



SUPPORT DE COURS SI4

HTML



FORMULAIRE

HTML 5

date	révision
Mars 2018	Création
12/10/2019	Ajout des 3 infographies + checkbox + radio...



TABLE DES MATIÈRES

1	Introduction.....	3
1.1	Les balises <FORM> et </FORM>.....	5
1.2	Balise input.....	5
1.2.1	Input type="text".....	5
1.2.2	Input type="submit" et input type="reset".....	6
1.2.3	input type="email", input type="url" et input type="tel".....	6
1.2.4	Input type="date", input type="time".....	7
1.2.5	Input type="button".....	8
1.2.6	Input type="password".....	8
1.2.7	input type="range".....	8
1.2.8	Input type="color".....	9
1.2.9	input type="checkbox".....	9
1.2.10	input type="radio".....	9
1.2.11	Input type="file".....	10
1.3	Balise Select.....	10
1.3.1	Liste déroulante.....	10
1.3.2	Ascenseur.....	11
1.4	Attribut TextArea.....	11
1.5	Attribut FieldSet.....	12
1.6	alert, confirm et prompt.....	12
1.7	Méthodes HTTP : GET et POST.....	13
1.7.1	Choix de la méthode.....	13
1.7.2	Envoi d'un courrier électronique.....	13
1.8	Pour aller plus loin.....	14
1.9	Quelles balises fonctionnent partout ?.....	14
2	Maîtriser les saisies.....	15
2.1	Les sélections de champs.....	15
2.1.1	Label.....	15
2.2	Les protections sur <Input type="texte">.....	16
2.2.1	Propriété : placeholder.....	16
2.2.2	Propriété : maxlength.....	16
2.2.3	Propriété : pattern.....	16

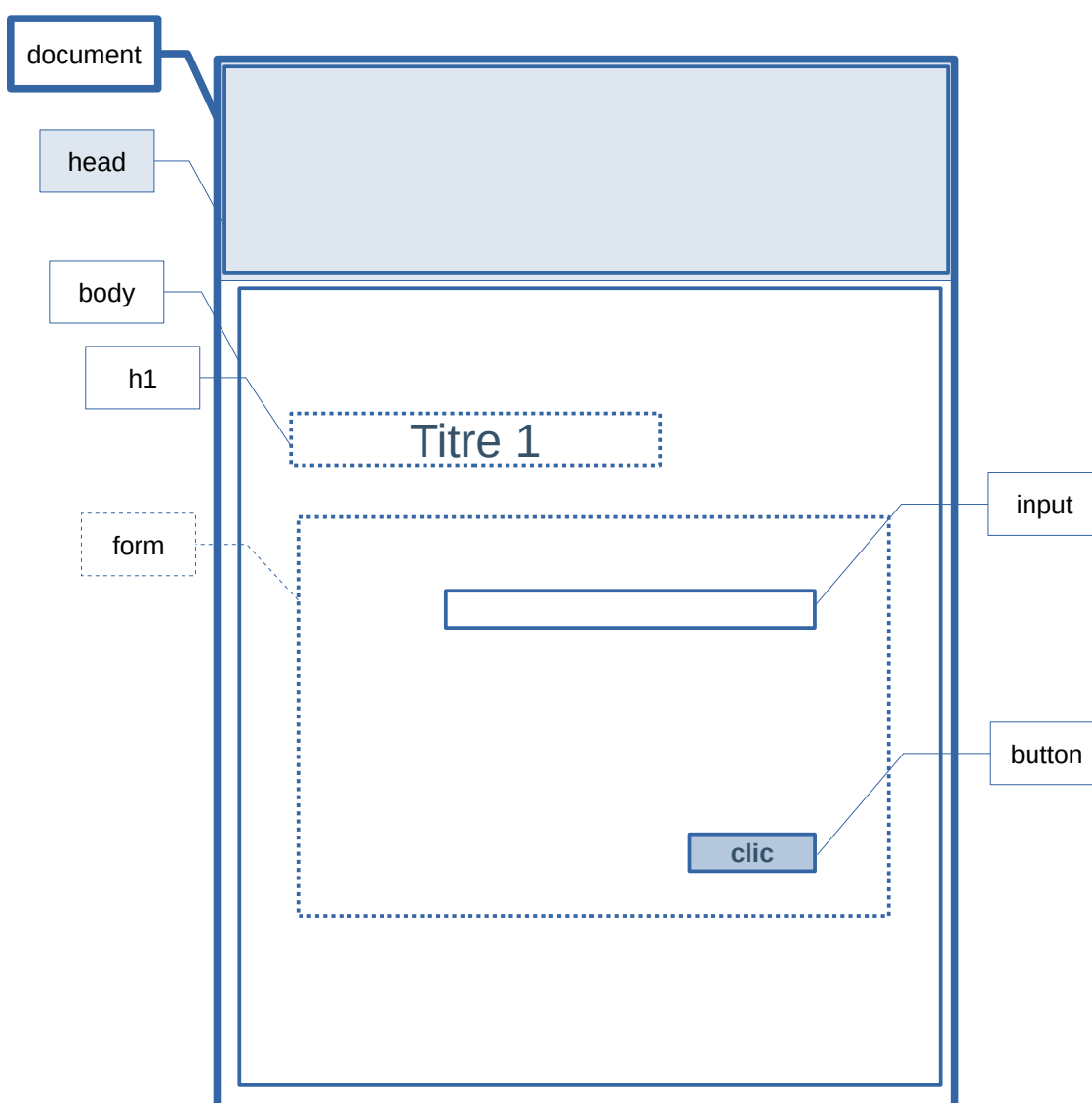
1 INTRODUCTION

Les formulaires font l'objet d'un support de cours particulier.

En effet, il ne s'agit ni plus, ni moins de codes HTML (et HTML 5), cependant, les formulaires sont les éléments de l'interface graphique en HTML qui permettent aux utilisateurs de transmettre des informations à JavaScript, PHP ou d'autres langages.

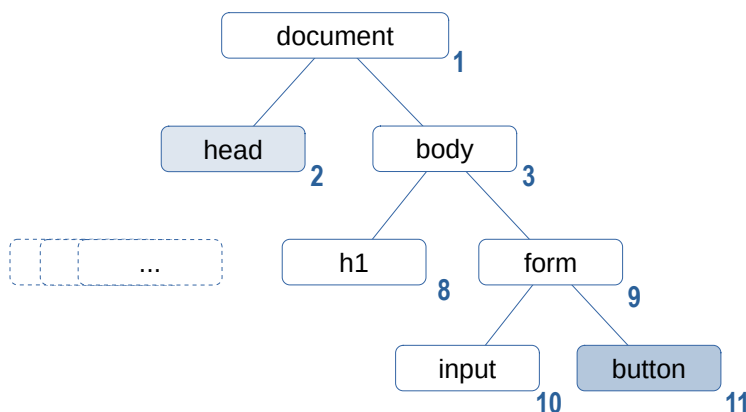
Il est donc nécessaire de bien connaître les possibilités pour limiter les erreurs de saisies des usagers.

Rappelons qu'une page HTML se situe dans un objet "document" du navigateur : il faudra en effet pouvoir agir avec les éléments du formulaire.



Ainsi, toutes les balises et tous les objets sont définis dans le DOM (Document Object Model), qui est une structure arborescente.

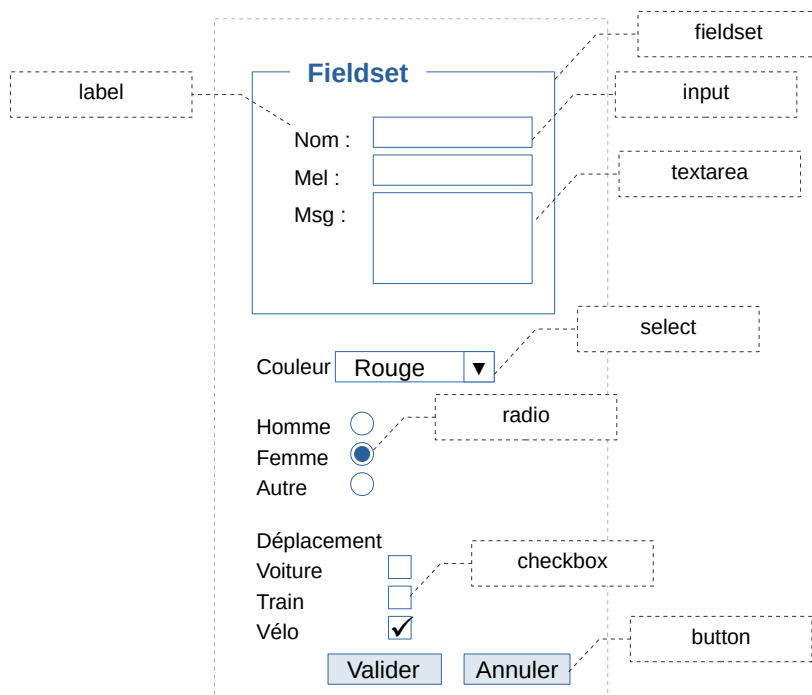
Pour rappel, un ordinateur est plus efficace dans le traitement des nombres que celui des chaînes. Il est donc logique de créer une structure basée sur les arbres et notamment les mathématiques avec la théorie des graphes.



La plupart des balises HTML sont simples mais les formulaires nécessitent des interactions et il existe donc de nombreuses balises dédiées pour gérer les saisies des utilisateurs. La plupart du temps, ces balises seront englobées à l'intérieur de balises `<form>` et `</form>` dont l'utilité est de pouvoir envoyer le contenu vers un serveur web.

Cependant, JavaScript pouvant exploiter la structure du DOM, il peut arriver que des balises de formulaire ne soient pas encadrées par les balises `<form>` `</form>`.

Les balises de formulaire les plus fréquentes sont les suivantes :



Le nombre de balises dans le document pouvant varier, il sera utile de pouvoir les repérer par un identifiant (ID). **Seule constante : il doit toujours y avoir un bouton de type "SUBMIT" pour soumettre le formulaire à un serveur web (ou un script Javascript).**



1.1 LES BALISES <FORM> ET </FORM>

Un formulaire, est un ensemble d'éléments contenus entre les balises <FORM> et </FORM>.

Il peut y avoir plusieurs formulaires sur une page, mais généralement, on en place qu'un seul ;

La balise <form> accepte plusieurs paramètres, et ci-dessous, voici les deux plus fréquents :

- **method** : qui accepte deux valeurs, GET et POST.
- **action** : qui accepte pour valeur, la page de script vers laquelle seront envoyées les données ou bien le lien (cela peut être une adresse e-mail, voir ci-après).

```
<form method="post" action="login.php">
...
  <input type="submit" value="Valider">
  <input type="reset" value="Annuler">
</form>
```

- Un formulaire doit être "envoyé" et le navigateur attend une action sur un bouton de type "submit". Le bouton de type "reset" vide tous les champs.

Il reste cependant possible de modifier l'aspect de ces balises en CSS et d'exploiter leurs contenus avec JavaScript.

1.2 BALISE INPUT

La balise <input> permet de gérer les saisies. Il existe plusieurs attributs pour la balise <input> qui déterminent les éléments utilisables.

NOTE IMPORTANTE : toutes les balises contenant des informations doivent avoir un nom : on ajoute l'attribut name avec comme valeur le nom. Exemple : name="nomPersonne".

1.2.1 Input type="text"

Cet attribut permet la saisie d'un texte sur une seule ligne.

```
<input type="text" name="nom"
placeholder="Totoski">
```

Identifiant :

Tableau 1.1 : balise input après le texte "Identifiant :"

La balise "name" désigne l'étiquette du champ de saisie.

La balise input accepte de multiples attributs très pratiques, dont **placeholder="TOTOSKI"** qui affiche un exemple de texte dans le cadre tant qu'il est vide, **autocomplete="on"** ou "off", **maxlength=nombre** (avec nombre qui est la valeur du nombre de caractères maximum), **required** qui impose que le champ ne soit pas vide, **autofocus** pour avoir le focus au chargement de la page, **readonly, disabled...**

1.2.2 Input type="submit" et input type="reset"

Ces deux attributs sont des boutons d'action dont il n'est pas possible de définir soi-même les actions.

Le premier bouton "submit" valide le formulaire et envoie les données au serveur.

Le deuxième "reset" réinitialise le formulaire (efface les champs).

<code><input type="reset"><input type="submit"></code>	
--	---

Tableau 1.2 : balise de type "reset" ou "submit"

1.2.3 input type="email", input type="url" et input type="tel"

Il s'agit de variantes du champ de saisie input="text". Deux avantages pour celles-ci :

- Un contrôle automatique de la saisie avant la validation
- Un affichage de clavier approprié sur les smartphones.

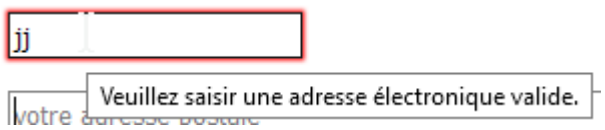
<code><input type="email"></code>	
---	--

Tableau 1.3 : balise de type "email"

Exemple du clavier de saisie lorsque ce champ est actif (focus).

Un certain nombre de touches ou symboles disparaissent et le symbole @ apparaît, afin de faciliter la saisie d'une adresse de courrier électronique.

Dans le cas de la balise input type="tel", le clavier numérique sera affiché.



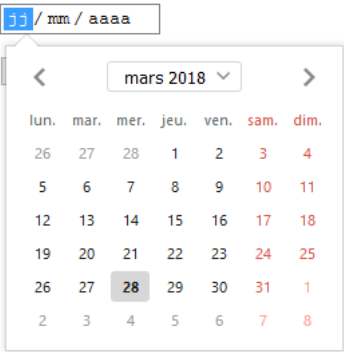
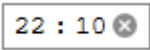
Il est possible de créer un champ type="text" dont les valeurs seront testées avec une expression régulière. Voici par exemple, un champ permettant la saisie d'un code postal (5 chiffres uniquement).

<code><input type="text" name="codepostal" maxlength=5 pattern="[0-9]{5}"></code>



1.2.4 Input type="date", input type="time"

Ce champ permet de contrôler la saisie d'une date, en proposant également l'affichage d'un calendrier.

<pre><input type="date"></pre>	
<pre><input type="time"></pre>	

D'autres valeurs sont parfois utilisables mais il faut surveiller la compatibilité avec l'ensemble des navigateurs.

Par exemple, au 28 mars 2018, les valeurs "datetime", "week", "month" ne sont pas encore reconnues partout.

IE	Edge *	Firefox	Chrome	Safari	iOS Safari *	Opera Mini *	Chrome for Android	UC Browser for Android	Samsung Internet
			49						
			63		10.2				
		58	64		10.3				4
11	16	59	65	11	11.2	all	64	11.8	6.2
	17	60	66	11.1	11.3				
		61	67	TP					
			68						

1.2.5 Input type="button"

Cet attribut permet de placer un bouton. La propriété associée est **value**="texte du bouton" pour écrire le texte dans le bouton. Cependant il est impératif de le lier à un événement Javascript.

```
<form>
  <input type="button" id="mybutton" value="dévalidez-moi">
</form>
<script>
  var btn = document.querySelector('#mybutton');
  btn.addEventListener('click', updateBtn);

  function updateBtn() {
    if (btn.value === 'dévalidez-moi') {
      btn.value = 'Je suis dévalidé';
    } else {
      btn.value = 'dévalidez-moi';
    }
  }
</script>
```

1.2.6 Input type="password"

Identique au champ de saisie, sa particularité est de ne pas afficher les caractères mais des étoiles ou des points.

```
<input type="password" name="UserPasswd">
```



Tableau 1.4 : balise de type "password"

1.2.7 input type="range"

Permet de récupérer une valeur numérique comprise dans la plage.

```
<input type="range" name="Volume" min="0"
max="100">
```



Tableau 1.5 : balise de type "range"

1.2.8 Input type="color"

Il s'agit d'un champ permettant de sélectionner une valeur RGB de couleur.

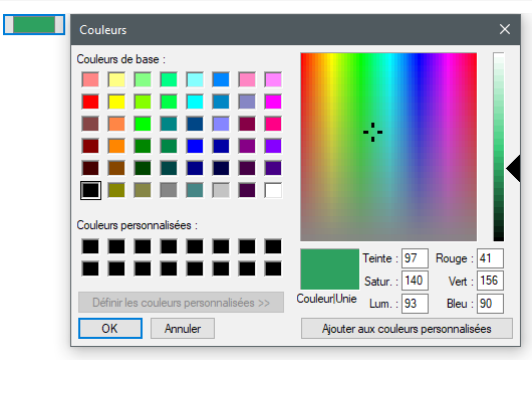
<pre><input type="color" /></pre>	
---	--

Tableau 1.6 : balise de type "color"

1.2.9 input type="checkbox"

Permet de proposer plusieurs choix simultanés.

<pre><input type="checkbox" name="rep1" value="Voiture"> j'utilise ma voiture
 <input type="checkbox" name="rep2" value="Train"> J'utilise plutôt le train
 <input type="checkbox" name="rep3" value="Vélo" checked> Je préfère utiliser le vélo
</pre>	<input type="checkbox"/> j'utilise ma voiture <input type="checkbox"/> J'utilise plutôt le train <input checked="" type="checkbox"/> Je préfère utiliser le vélo
--	--

Tableau 1.7 : balise de type "checkbox"

1.2.10 input type="radio"

Sur les choix proposés, un seul peut être sélectionné à la fois (l'attribut "name" doit être identique).

<pre><input type="radio" name="sexe" value="H"> Homme
 <input type="radio" name="sexe" value="F"> Femme
 <input type="radio" name="sexe" value="N" checked> Non- défini
</pre>	<input type="radio"/> Homme <input type="radio"/> Femme <input checked="" type="radio"/> Non-défini
---	---

Tableau 1.8 : balise de type "radio"

1.2.11 Input type="file"

Il existe de nombreux autres type pour les attributs <input>. Une liste non-exhaustive est indiquée ci-contre.

L'attribut 'file' permet l'envoi d'un fichier par exemple :

```
<form method="post" enctype="multipart/form-data" action="script">
  <input type="file" name="img" accept="image/*" />
  <input type="file" name="docWd" accept=".docx" />
</form>
```

1.3 BALISE SELECT

La balise SELECT permet de créer des listes (déroulantes ou ascenseurs).

1.3.1 Liste déroulante

Il suffit d'appeler l'attribut select puis d'utiliser la propriété <option> autant de fois qu'il y a d'éléments dans la liste.

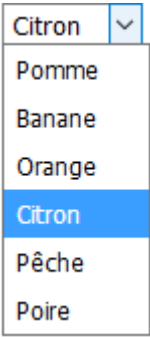
<pre><select Name="menu"> <option> Pomme</option> <option> Banane</option> <option> Orange</option> <option selected> Citron</option> <option> Pêche</option> <option> Poire</option> </select ></pre>	
--	---

Tableau 1.9 : exemple d'utilisation select en mode liste

Il est possible de récupérer une valeur numérique en ajoutant la propriété valeur dans les options :

```
<option value="1"> Pomme</option>
```

1.3.2 Ascenseur

Pour obtenir une liste avec ascenseur et sélection multiple, il suffit de rajouter la taille du nombre d'éléments affichés simultanément et la propriété 'multiple'

```
<select Name="menu[]" size=4 multiple>
  <option> Pomme</option>
  <option> Banane</option>
  <option selected> Orange</option>
  <option selected> Citron</option>
  <option> Pêche</option>
  <option> Poire</option>
</select >
```



Tableau 1.10 : exemple d'utilisation select en mode ascenseur

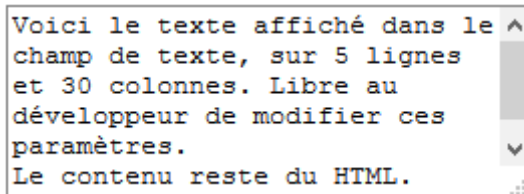
Notez les crochets dans le nom menu[] : pour récupérer les attributs sélectionnés sous forme d'un tableau.

1.4 ATTRIBUT TEXTAREA

C'est un attribut pour créer un champ de texte. Son fonctionnement est simple : on définit les lignes et colonnes et on peut indiquer le texte contenu initialement (l'utilisateur peut l'effacer).

```
<textarea rows="5" cols="30">Voici le texte affiché dans le champ de texte, sur 5
lignes et 30 colonnes. Libre au développeur de modifier ces paramètres.&#13;Le
contenu reste du HTML.<
</textarea>
```

Notez simplement la présence du code `` qui impose un retour chariot dans le texte.



La propriété **placeholder** fonctionne sur ce type de saisie, cela reste plus propre.

1.5 ATTRIBUT FIELDSET

Plutôt esthétique, cet attribut permet de rendre l'interface du formulaire plus claire, en encadrant un ensemble de balises.

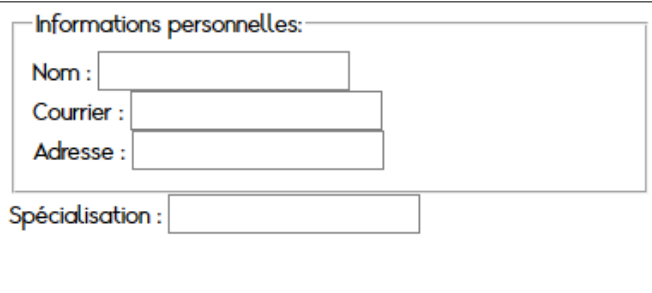
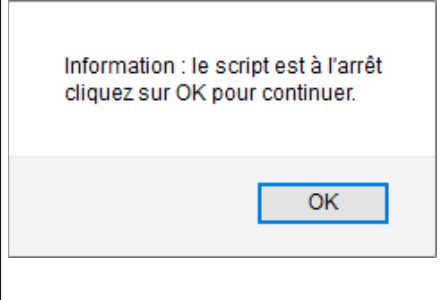
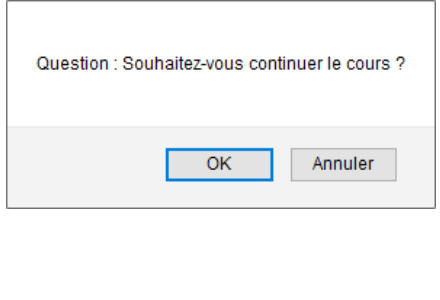
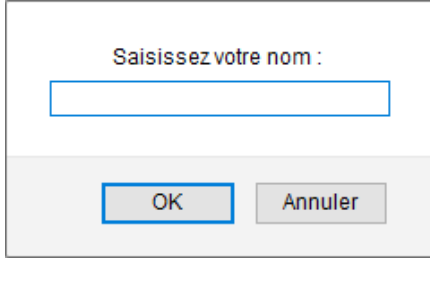
<pre><form> <fieldset> <legend>Informations personnelles:</legend> Nom : <input type="text" name="nom">
 Courrier : <input type="text" name="courrier">
 Adresse : <input type="text"> </fieldset> Spécialisation : <input type="text" name="slamsisr">
 </form></pre>	
---	--

Tableau 1.11 : utilisation d'un fieldset pour délimiter des informations

1.6 ALERT, CONFIRM ET PROMPT

Il existe également une petite famille d'éléments utilisables avec les utilisateurs. Ce sont des boîtes de dialogue qui affichent ou demandent un message :

window.alert(chaine)	window.confirm(chaine)	window.prompt(chaine)
		
<p>Aucune valeur de retour. Un seul bouton. <i>Pour les messages longs, utiliser \n pour passer à la ligne.</i></p>	<p>Retour booléen (OK = vrai) Deux boutons</p>	<p>Retourne une chaîne. <i>Accepte une chaîne par défaut en paramètre supplémentaire prompt("votre nom ?", "Bob")</i></p>

À noter : pour les 3 méthodes, il n'est pas obligatoire de mettre le mot clé "window".



1.7 MÉTHODES HTTP : GET ET POST

L'envoi d'un formulaire (et donc des valeurs de chaque balise le composant) se fait avec une balise `<input="submit">` et le choix de l'action dans la balise `<form>`.

1.7.1 Choix de la méthode

Utiliser une méthode GET transmet les données dans le lien tandis qu'une méthode POST envoie les données dans un tableau caché dans le corps de la requête du navigateur.

Les deux méthodes ont des avantages et inconvénients mais POST l'emporte pour la sécurité :

- Le protocole **HTTP-GET** crée une chaîne des paires nom-valeur et ajoute la chaîne à la suite de l'URL. Les données sont donc visibles dans la barre de lien. De plus, la longueur est limitée à 2048 caractères (et uniquement en ASCII).
- Le protocole **HTTP-POST** crée aussi une chaîne des paires nom-valeur, mais elle est transmise dans le corps du message (de la requête HTTP). Elle n'est plus visible et peut transporter plus de caractères (et également des données binaires).

GET sera plus utilisé pour accéder à une page dont les paramètres ne sont pas secrets.

1.7.2 Envoi d'un courrier électronique

Il est aussi possible de générer un courrier électronique qui lancera le client de messagerie de l'utilisateur (Outlook, Thunderbird... mais pas de webmail) avec les éléments prédéfinis :

```
<FORM Action="mailto:david.roumanet@ac-grenoble.fr" METHOD="POST" ENCTYPE="text/plain">
  Formulaire d'envoi email :
  <Br>Sujet:
  <INPUT name="Subject" value="Exemple de sujet bidon">
  <Br>Corps:&#xa0;
  <TEXTAREA name="Body">texte dans le corps du mail.</TEXTAREA>
  <BR>
  <INPUT type="submit" value="Submit">
</FORM>
```

La méthode utilisée `METHOD="GET"` ajoute des caractères '+', tandis que `METHOD="POST"` sera plus cohérent mais les champs seront placés dans le corps du courrier.



1.8 POUR ALLER PLUS LOIN...

Je vous recommande de lire la page web suivante :

<https://openclassrooms.com/courses/apprenez-a-creer-votre-site-web-avec-html5-et-css3/les-formulaires-8>

1.9 QUELLES BALISES FONCTIONNENT PARTOUT ?

Le nombre de commandes et balises en HTML augmente régulièrement et la tentation peut être forte d'utiliser les dernières nouveautés. Cependant, tout le monde n'utilise pas le même navigateur, ni même la même version !

En entreprise, il est fréquent que les versions de navigateurs soient bridées pour rester compatibles avec des outils métiers internes. Un classique est l'ERP SAP/R3 par exemple. Utilisez le site <https://caniuse.com/> pour vérifier si une commande est complètement valide sur l'ensemble des navigateurs.



2 MAÎTRISER LES SAISIES

L'ensemble des éléments qui peuvent constituer un formulaire ont été vus au chapitre précédent. Ce chapitre s'intéresse au bon usage des éléments, pour sécuriser les saisies :

- L'utilisateur peut ne pas comprendre l'usage d'un champ
- L'utilisateur peut se tromper lors de la saisie (nombre de caractères, types de caractères, etc.)
- L'utilisateur peut tester les différentes possibilités pour "voir" (comportement de hacker)
- L'utilisateur peut être un robot programmé pour altérer les données du site

Il faut donc pouvoir réduire les possibilités d'erreur.

2.1 LES SÉLECTIONS DE CHAMPS

La plupart des champs doivent être précédés d'une description. Ce petit texte est appelé "label" sur les éditeurs d'interfaces graphiques.

2.1.1 Label

En HTML, une bonne pratique consiste à créer un label pour chaque champ et de les associer entre-eux par leur ID. L'intérêt est double, car il est possible d'affecter un style CSS à cette balise et ainsi améliorer l'affichage du formulaire, mais aussi de cliquer sur le label pour sélectionner le champ de saisie.

```
<label for="IDadr">Adresse : </label><input type="text" name="adr" id="IDadr" />
```

Il est obligatoire que le contenu de l'attribut for dans la balise "label" soit le même que la valeur de l'ID dans la balise "input".

Par la suite, il est possible de créer un style CSS pour les labels qui devra être un bloc flottant pour s'afficher à côté du champ de saisie, tout en ayant une largeur définissable (par défaut, "label" est une balise %INLINE)

Exemple de code CSS :

```
label { display:block; width:80px; float:left; color:blue; }
```



2.2 LES PROTECTIONS SUR <INPUT TYPE="TEXTE">

C'est l'élément le plus utilisé, il dispose de nombreuses propriétés et peut ainsi éviter de nombreuses erreurs de saisie

2.2.1 Propriété : placeholder

La propriété "placeholder" permet d'afficher un texte grisé dans le champ, mais ce texte ne sera pas retourné lors de la soumission du formulaire. Dès que l'utilisateur saisit un caractère, ce texte disparaît... mais réapparaîtra dès que le champ est de nouveau vide.

On utilise donc placeholder comme une phrase de conseil ou d'aide.

```
<form action="script.php" method="post">  
Code postal : <input type="text" placeholder="entrez un code sur 5 chiffres"><BR>  
<input type="submit">
```

2.2.2 Propriété : maxlength

La propriété "maxlength" limite le nombre de caractères saisis. Une fois cette limite atteinte, il n'est plus possible de continuer la saisie, même en copiant un texte plus long, la saisie sera tronquée aux premiers caractères.

```
<form action="script.php" method="post">  
Code postal : <input type="text" placeholder="entrez un code sur 5 chiffres"  
                maxlength=5><BR>  
<input type="submit">
```

2.2.3 Propriété : pattern

La propriété "pattern" est très puissante mais exige de connaître les expressions régulières. Un support de cours particulier est fourni.

Voici cependant un exemple simple (toujours sur le code postal) :

```
<form action="script.php" method="post">  
Code postal : <input type="text" placeholder="entrez un code sur 5 chiffres"  
                maxlength=5 pattern="[0-9]{5}"><BR>  
<input type="submit">
```

Ici, on indique que les caractères doivent être dans l'intervalle de 0 à 9 et qu'il doit y en avoir exactement 5.

La propriété ne bloque pas au moment de la saisie mais lorsque l'utilisateur quitte le champ, celui-ci devient rouge. Dans notre cas, on ne bloque pas la saisie de caractères alphabétiques, mais on signale l'erreur à la sortie du champ.